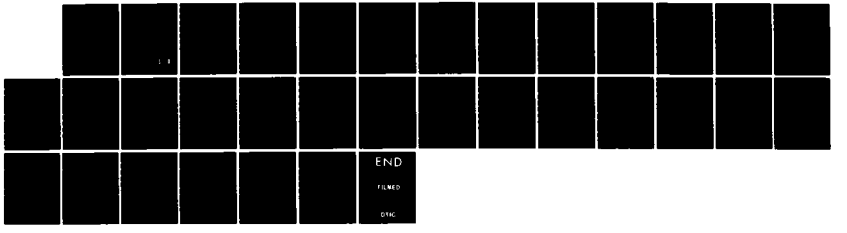
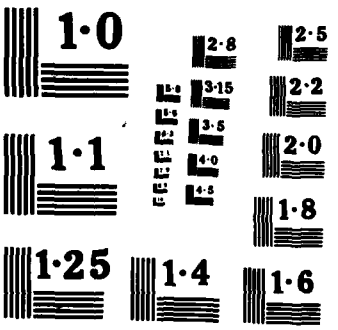
AD-A152 523 DEFENSE DATA NETWORK SERVICE ACCESS PROTOCOLS(U) MITRE CORP MCLEAN VA MITRE C3I DIV R K MILLER FEB 84 MTR-84W00005 F19628-84-C-0001 1/1

UNCLASSIFIED F/G 15/3 NL




1·0
1·1
1·25
1·4
2·8
3·15
3·5
4·0
4·5
2·5
2·2
2·0
1·8
1·6

(1)

Defense Data Network
Service Access Protocols

MITRE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A152323 — 3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* DEFENSE DATA NETWORK SERVICE ACCESS PROTOCOLS	5. TYPE OF REPORT & PERIOD COVERED
	6. PERFORMING ORG. REPORT NUMBER MTR-84W00005
7. AUTHOR(s) Robert K. Miller, Jr.	8. CONTRACT OR GRANT NUMBER(s) F19628-84-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS The MITRE Corporation 1820 Dolley Madison Blvd. McLean, Virginia	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS (Sponsor) Defense Communications Agency/Defense Data Network Program Management Office	12. REPORT DATE February 1984
	13. NUMBER OF PAGES 26
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*	15. SECURITY CLASS *(of this report)* U
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT *(of this Report)* Approved for public release; distribution is unlimited.	Accession For NTIS GRA&I [X] DTIC TAB [ ] Unannounced [ ] Justification
17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*	By Distribution/ Availability Codes
18. SUPPLEMENTARY NOTES	Dist | Avail and/or Special A-1

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Service access Protocols, Protocols, access control, Protocol specifications, Service access, Transmission Control Protocol (TCP), Internet Protocol (IP)

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This document contains Service Access Protocol (SAP) specifications, as defined in the Defense Data Network (DDN) Host Front-end Protocol (HFP) specification. The "service access" layer is the uppermost layer of the three layers that make up the HFP. The service access layer is responsible for the interpretation of the commands and the responses that are exchanged between a "host" and a front-end device to manipulate the peer-to-peer protocols that have been offloaded to the front-end.

DD FORM 1 JAN 73 1473 EDITION OF NOV 65 IS OBSOLETE

UNCLASSIFIED

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

Defense Data Network Service Access Protocols

Robert K. Miller, Jr.

February 1984

MTR-84W00005

SPONSOR:
Defense Communications Agency/Defense Data Network
Program Management Office
CONTRACT NO.:
F19628-84-C-0001

The MITRE Corporation
MITRE C³I Division
Washington C³I Operations
1820 Dolley Madison Boulevard
McLean, Virginia 22102

ABSTRACT

This document presents the protocol specifications for the service access layer of the Defense Data Network (DDN) Host Front-end Protocol (HFP). The service access layer is responsible for the interpretation of the commands and the responses that are exchanged between a host and the Host Front End Processor (HFEP) configuration of the DDN Network Access Component. Specifications are defined to support communications with the Transmission Control Protocol (TCP) and the Internet Protocol (IP) implementations in the HFEP.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

1.0 INTRODUCTION

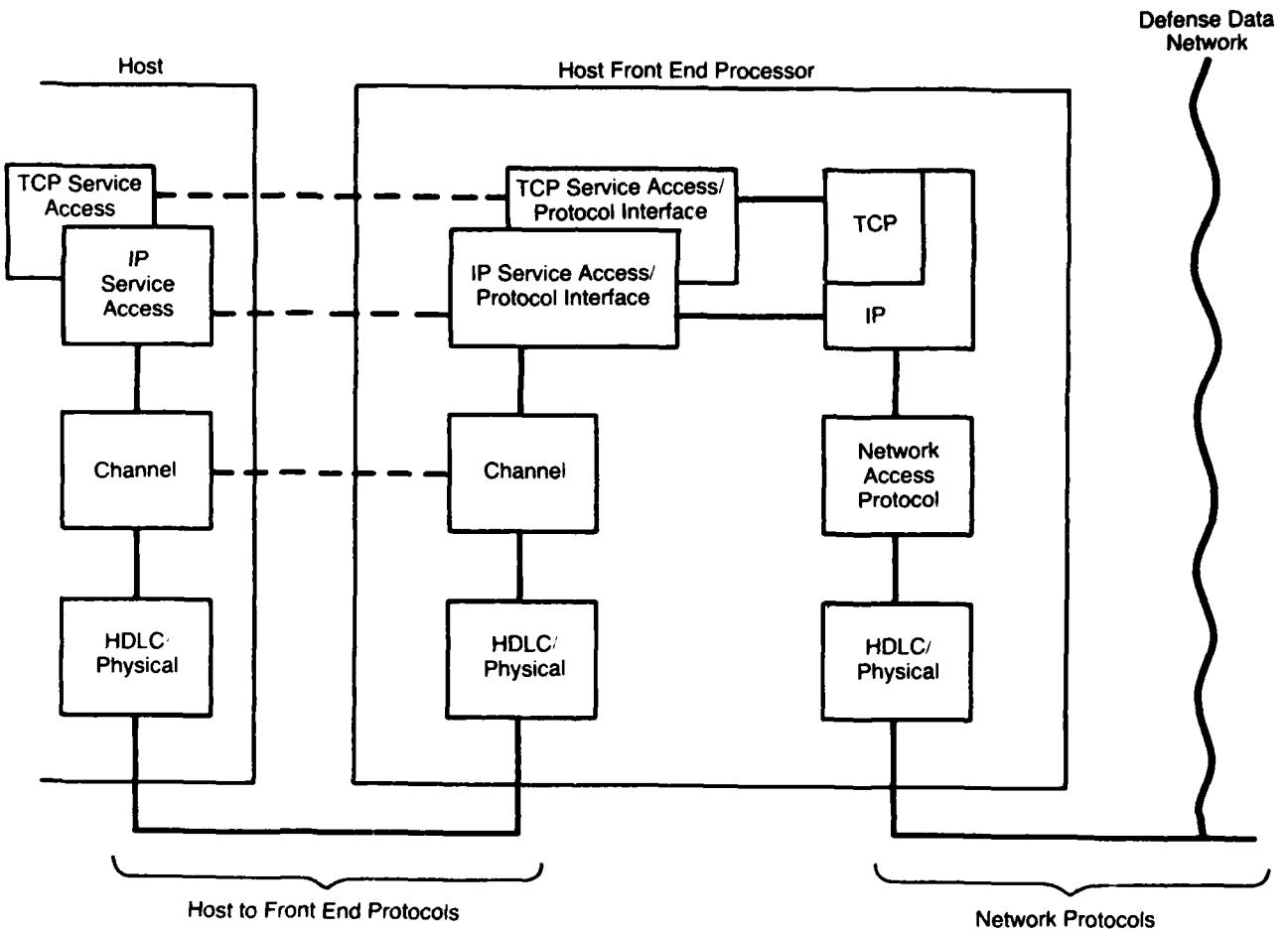
This document contains Service Access Protocol (SAP) specifications, as defined in the Defense Data Network (DDN) Host Front-end Protocol (HFP) specification.[1] The "service access" layer is the uppermost layer of the three layers that make up the HFP. The service access layer is responsible for the interpretation of the commands and the responses that are exchanged between a "host" and a front-end device to manipulate the peer-to-peer protocols that have been offloaded to the front-end. The layering of the HFP is shown in Figure 1.

1.1 Service Access Protocol Definition

The HFP specification defines a Service Access Protocol as follows:

> Each Service Access Protocol is described by a Service Access Protocol specification and a set of Service Access Protocol adaptation descriptions. A Service Access Protocol specification defines the *rules for communication between* apposite SAPIs [Service Access Protocol Interpreters] in the host and in the front-end. The unit of service access communication is the Service Access Protocol message. Service Access Protocol messages correspond to Channel Protocol messages, i.e., there is a Service Access Protocol begin_command corresponding to the Channel Protocol Begin_Command, a Service Access Protocol transmit_command corresponding to the Channel Protocol Transmit_Command, etc. (To distinguish Service Access Protocol messages from Channel Protocol messages, Service Access Protocol messages are written all lower case.) Service Access Protocol messages are carried by the TEXT field of the corresponding Channel Protocol messages. Thus, specifying a Service Access Protocol amounts to defining the TEXT fields of the Channel Protocol messages in terms of the corresponding Service Access Protocol messages.
>
> Since choices must be made in implementing a Service Access Protocol for a given host and front-end (e.g., in handling mismatch between host and front-end word sizes and/or character sets), an adaptation description describing these

Figure 1
Host Front End Protocol Architecture



choices must also be made for each implementation of the Service Access Protocol.

The Service Access Protocol specifications defined in this document are intended for use with the Host Front End Processor (HFEP) configuration of the DDN Network Access Component (NAC).[2] Specifications are derined to support communication with the Transmission Control Protocol (TCP) and the Internet Protocol (IP) implementations in the HFEP. A Service Access Protocol specification for the HFP maintenance service that provides the management functions for the Channel Protocol has been defined in the HFP specification.

1.2 Text Field Conventions

Normally implementation details are reserved for the SAP adaptation descriptions. However, since the DDN Host Front End Processor is being designed to support a standard interface, the following TEXT field conventions have been adopted. TEXT in the Channel Protocol messages consists of fields of varying length. Each field will consist of one or more data octets. Bits within a field will be numbered from left to right. For example, if the field consists of one octet, the lert (most significant) bit will be designated bit number zero, and the right (least significant) bit will be designated bit number seven. The contents of each field will be interpreted either as a right justified, 2's complement binary integer or a series of flag bits. Flag bits will be allocated from left to right within a field. Unless otherwise specified, the value zero (0) will be used to designate "don't care".

2.0 TCP SERVICE ACCESS PROTOCOL SPECIFICATION

2.1 HFP Code Number

The HFP Code Number for the TCP service is two (2). (Note: the HFP Code Number is used in the Service field of the Begin_Command of the Channel Protocol to specify the SAPI to which the channel is being established.)

2.2 Description of the Service

This specification defines the Service Access Protocol to access the HFEP implementation of the DoD standard Transmission Control Protocol, MIL-STD-1778.[3] The TCP and its associated Internet Protocol, MIL-STD-1777,[4] provide reliable host-to-host peer level communication for the Defense Data Network.

The TCP Service Access Protocol provides the interface between a "client" process and the implementation of the TCP in the HFEP configuration of the DDN Network Access Component. The phrase "client" process refers to a process operating within a subscriber device, e.g., a host computer system, that is connected to a NAC and supports communications via the HFP. The TCP Service Access Protocol is implemented in both the client device and the NAC. Throughout this specification, "client process" will be used to refer to the portion of the SAP that is implemented in a subscriber device, and "service module" will be used to refer to the portion of the SAP that is implemented in the HFEP. The client process and the service module use the HFP Channel Protocol messages to exchange information. A synopsis of specific message types used follows.

begin command

> sent by the client process to request that a connection be established or to listen for a connection establishment request from the remote TCP

sent by the service module to initiate communications between the HFEP and the Host in response to a connection establishment request from the remote TCP.

begin response

sent by the client process or the service module to confirm the establishment of a connection or report the reason for failure

end command

sent by the client process to request that a connection be closed

sent by the service module to report the closing of a connection by the remote TCP

end response

sent by the client process or the service module to acknowledge the closing of a connection

execute command

sent by the client process to request that a connection be aborted or to request the status of a connection

sent by the service module to report a connection abort by the remote TCP

execute response

sent by the client process to acknowledge that a connection was aborted

sent by the service module to acknowledge that a connection was aborted or to report the status of a connection

transmit command

sent by the client process to transfer data to the service module for transmission to the remote TCP

sent by the service module to transfer data to the client process that was received from the remote TCP

The details of each message type are described in the following section.

2.3 Message Use

2.3.1 begin command

2.3.1.1 When Sent. A begin_command is sent by the client process to request that the service module attempts to establish a TCP connection or to listen for a TCP connection establishment request from the remote TCP. A begin_command is sent by the service module to initiate communications between the HFEP and the Host in response to a connection establishment request from the remote TCP.

2.3.1.2 Action on Receipt. When the client process receives the begin_command, it will notify the upper layer protocol to initiate communications and will send a begin_response to confirm the establishment of communications with the host. When the service module receives the begin_command, it will attempt to establish a connection or to listen for a connection based on the Active/Passive flag in the TEXT field. Following notification by the local TCP, the service module will send a begin_response to report the outcome of establishment processing.

2.3.1.3 TEXT Field Syntax.

Local Host:	FIXED(32)
Foreign Host:	FIXED(32)
Local Port:	FIXED(16)
Foreign Port:	FIXED(16)
Timeout:	FIXED(16)
Type of Service:	FIXED(8)
Options:	VARIABLE(?)

(Note: format details for TEXT field syntax are described in the HFP specification.[1])

2.3.1.4 TEXT Field Semantics.

a. Local Host. This field specifies the local host address for the connection. Depending on the configuration, assignment of a value to this field may be optional.

b. Foreign Host. This field specifies the foreign host address for the connection. If the Active/Passive flag is set to passive, the local TCP will listen for the host specified in this field. If this field is set to zero, the local TCP will wait for a call from any host.

c. Local Port. This field specifies the local port number to be used to establish a connection. If this field is set to zero, the local TCP will select a local port number for the connection.

d. Foreign Port. This field specifies the foreign port number for the connection. If the Active/Passive flag is set to passive, the local TCP will listen for the port specified in this field. If this field is set to zero, the local TCP will wait for a call from any port.

e. Timeout. This field specifies the maximum time, in seconds, that the client process is willing to wait for an acknowledgement from the remote TCP. If data is not successfully delivered within the timeout period, the local TCP will abort the connection. The default value is 65535. The timeout mechanism is described in Paragraph 9.2.9 of MIL-STD-1778.

f. Type of Service. This field specifies the IP Type of Service field, as described in Paragraph 9.3.3 of MIL-STD-1777. TCP precedence considerations are described in Paragraph 9.2.11 of MIL-STD-1778. Bit 7 of this field is modified to contain the Active/Passive flag. This flag specifies the type of request. If the flag is set to active (bit is zero), the begin_command is an establishment request for the local TCP. If the flag is set to passive (bit is one), the begin_command is a request for the local TCP to listen for a connection from the remote TCP. The default value is Active. Active/Passive open mechanisms are described in Paragraph 9.2.13 of MIL-STD-1778. When the service module issues a begin_command, this flag must be set to Active.

g. Options. This field specifies the TCP and IP options to be used. The format and definition of these options are

described in Paragraph 9.3.11 of MIL-STD-1778 and Paragraphs 9.3.13 and 9.3.15 of MIL-STD-1777, respectively.

2.3.2 begin response

2.3.2.1 When Sent. A begin_response is sent by the client process or the service module to report the outcome of an attempt to establish a TCP connection or to listen for a TCP connection that was requested by a begin_command.

2.3.2.2 Action on Receipt. When the client process or the service module receives a successful connection indication, it may proceed to transmit and receive data over the connection. When the client process or the service module receives an unsuccessful connection indication, it should perform appropriate error recovery processing.

2.3.2.3 TEXT Field Syntax.

Local Connection Name:	FIXED(16)
Result:	FIXED(8)

2.3.2.4 TEXT Field Semantics.

a. Local Connection Name. This field will contain the local connection name, described in Paragraph 6.5.1.1 of MIL-STD-1778, used to identify the TCP connection.

b. Result. This field contains an encoding of the result of the begin_command. The codes are:

0 The command was successful.

1 A syntax error was detected in the TEXT field.

2 The security specified is not allowed for this connection.

3 The precedence specified is not allowed for this connection.

4 The local TCP does not have sufficient resources to establish a connection.

5 The connection that was requested to be opened already exists.

7 The connection that was requested to be opened is currently being terminated.

8 A system failure has caused the connection to reset.

9 A service failure by the HFP exists such that no data can be exchanged.

10 An error condition exists in the lower level protocols.

11 The user timeout has been exceeded.

2.3.3 end command

2.3.3.1 When Sent. An end_command is sent by the client process to request the service module to close a TCP connection. An end_command is sent by the service module to report to the client process that a connection has been closed by the remote TCP. The end_command will be processed as a non-flushing termination by the Channel Protocol. (Note: non-flushing termination is specified by setting the Flush away bit to zero in the the Control field. Queued data awaiting delivery will be sent prior to the initiation of termination processing. Termination mechanisms are described in the HFP Specification.)

2.3.3.2 Action on Receipt. When the client process receives an end_command, it should send an end_response and initiate appropriate termination action. When the service module receives an end_command, all data transmissions should be completed prior to initiating the connection closing. When the connection has been closed, the service module will send an end_response.

2.3.3.3 TEXT Field Syntax.

Local Connection Name: FIXED(16)

2.3.3.4 TEXT Field Semantics.

a. Local Connection Name. This field specifies the local connection name of the connection to be closed or the connection that has been closed by the remote TCP. There is no default value for this field.

2.3.4 end response

2.3.4.1 When Sent. The end_response is sent by the client process or the service module to acknowledge the closing of a connection.

2.3.4.2 Action on Receipt. When an end_response is received by the client process or the service module, the connection should be considered to be closed and the HFP logical channel to be terminated.

2.3.4.3 TEXT Field Syntax.

Local Connection Name: FIXED(16)
Result: FIXED(8)

2.3.4.4 TEXT Field Semantics.

a. Local Connection Name. The contents of this field will remain unchanged from the corresponding end_command.

b. Result. This field contains an encoding of the result of the end_command. The codes are:

0 The command was successful.

1 A syntax error was detected in the TEXT field.

6 The connection that was requested to be terminated does not exist.

7 The connection that was requested to be terminated is currently being terminated.

2.3.5 execute command

2.3.5.1 When Sent. The execute_command is sent by the client process to request that a connection be aborted or to request the status of a connection. The execute_command is sent by the service module to advise the client process that the connection was aborted by the remote TCP or that the remote TCP or the network has failed. (Note: An abort request is sent with the Synchronize bit in the Control field of the Channel Protocol message set to zero and the Attention bit set to one. A status request is sent with the Synchronize bit set to one and the Attention bit set to zero. Synchronization mechanisms are described in the HFP specification.)

2.3.5.2 Action on Receipt. When the client process receives an execute_command, it should send an execute_response and initiate appropriate recovery action. When an execute_command is received by the service module, the service module will initiate processing based on the type of request. Abort messages will be delivered in an expedited fashion, whereas status messages will be synchronized. If the Abort/Status flag is abort, the service module will initiate the processing to abort the connection. When the connection has been aborted, the service module will send an execute_response. If the Abort/Status flag is status, the service module will initiate a TCP status request. Upon completion of the request, the service module will send the status data to the client process via an execute_response.

2.3.5.3 TEXT Field Syntax.

Local Connection Name:	FIXED(16)
Abort/Status:	FIXED(8)

2.3.5.4 TEXT Field Semantics.

a. Local Connection Name. This field specifies the local connection name of the connection to be aborted or to obtain status information from. There is no default value for this field.

b. Abort/Status. This flag specifies the type of request. If the flag is set to abort (bit is zero), the execute_command is an abort request. If the flag is set to status (bit is one), the execute_command is a status request. The default value is abort.

2.3.6 execute response

2.3.6.1 When Sent. An execute_response is sent by the client process or the service module to report the outcome of an execute_command.

2.3.6.2 Action on Receipt. When a client process receives an execute_response, it should examine the Abort/Status field and the Result field to determine the appropriate processing to be initiated. Status Data will be returned only on a status request. When the service module receives an execute_response, it should consider the HFP logical channel to be terminated.

2.3.6.3 TEXT Field Syntax.

```
Local Connection Name:      FIXED(16)
Abort/Status:               FIXED(8)
Result:                     FIXED(8)
Status Data:                COMPLEX(?)
   Local Host:                 FIXED(32)
   Foreign Host:               FIXED(32)
   Local Port:                 FIXED(16)
   Foreign Port:               FIXED(16)
   Unacknowledged Data:        FIXED(16)
   Unreceived Data:            FIXED(16)
   Send Window:                FIXED(8)
   Receive Window:             FIXED(8)
   Connection State:           FIXED(8)
   Urgent Mode:                FIXED(8)
```

Timeout:	FIXED(8)
Type of Service:	FIXED(8)
Security:	FIXED(72)

a. Local Connection Name. The contents of this field will remain unchanged from the corresponding execute_command.

b. Abort/Status. The contents of this flag will remain unchanged from the corresponding execute_command.

c. Result. This field contains an encoding of the result of the execute_command. The codes are:

0 The command was successful.

1 A syntax error was detected in the TEXT field.

6 The connection that was specified does not exist.

d. Local Host. This field will contain the local host address for the connection.

e. Foreign Host. This field will contain the foreign host address for the connection.

f. Local Port. This field will contain the local port number for the connection.

g. Foreign Port. This field will contain the foreign port number for the connection.

h. Unacknowledged Data. This field will contain the number of octets of data for which the local TCP is currently awaiting acknowledgement.

i. Unreceived Data. This field will contain the sequence number of the next data octet expected to be received for the connection from the local TCP's viewpoint. This field will contain the number of octets of data currently pending receipt by the local TCP.

j. Send Window. This field will contain the allowed number of octets that may be sent to the remote TCP relative to any unacknowledged data.

k. Receive Window. This field will contain the allowed number of octets to be received from the remote TCP relative to the next expected data octet.

l. Connection State. This field will contain an encoding of the current state of connection from the local TCP's viewpoint. The TCP entity states are:

0 CLOSED. The connection does not exist.

1 LISTEN. The local TCP is waiting for a connection request from a remote TCP.

2 SYN RECVD. The local TCP is waiting for an acknowledgment after having both received and sent a connection request.

3 SYN SEND. The local TCP is waiting for a matching connection request after having sent a connection request.

4 ESTAB. The connection is established.

5 FIN WAIT1. The local TCP is waiting for a connection termination request from the remote TCP or an acknowledgment of the connection termination request previously sent.

6 FIN WAIT2. The local TCP is waiting for a connection termination request from the remote TCP.

7 CLOSE WAIT. The local TCP is waiting for a connection termination request from the local user (i.e., the service module).

8 CLOSING. The local TCP is waiting for a connection termination request acknowledgment from the remote TCP.

9 LAST ACK. The local TCP is waiting for an acknowledgment of the connection termination request that was previously sent to the remote TCP.

10 TIME WAIT. The local TCP is waiting for enough time to pass to ensure that the remote TCP has received the acknowledgment of its connection termination request.

m. Urgent Mode. This flag will indicate that the local process (i.e., the service module) should go into "urgent mode." The

urgent mechanism is described in Paragraph 9.2.8 of MIL-STD-1778.

n. Timeout. This field will contain the maximum delay allowed for data transmitted on the connection.

o. Type of Service. This field will contain the current IP Type of Service setting for the connection.

p. Security. This field will contain the current IP Security setting for the connection.

2.3.7 transmit command

2.3.7.1 When Sent. The transmit_command is sent by the client process to transfer data to the service module for transmission over a connection to the remote TCP. The transmit_command is sent by the service module to transfer data to the client process that was received from the remote TCP. Data may be sent only when the connection is in the established state and when the HFP flow control discipline permits. (Note: Flow control mechanisms are described in the HFP specification.)

2.3.7.2 Action on Receipt. When the client process receives a transmit_command, it should initiate processing of the data received from the remote TCP. When the service module receives a transmit_command, it will initiate the processing to transmit the data over the network.

2.3.7.3 TEXT Field Syntax.

Data Count:	FIXED(32)
Local Connection Name:	FIXED(16)
Flags:	
Push	
Urgent	
Data:	VARIABLE(?)

2.3.7.4 TEXT Field Semantics.

a. Data Count. This field specifies the size, in octets, of the data being transmitted. The default value is zero.

b. Local Connection Name. This field specifies the local connection name or the connection over which data should be transmitted. There is no default value for this field.

c. Push Flag. This flag (i.e., the bit set to one) specifies that the TCP push service is being requested. The default value is zero.

d. Urgent Flag. This flag (i.e., the bit set to one) specifies that the TCP urgent service is being requested. The default value is zero.

e. Data. This field contains the TCP data being transmitted.

3.0 IP SERVICE ACCESS PROTOCOL SPECIFICATION

3.1 HFP Code Number

The HFP Code Number for the IP service is one (1). (Note: the HFP Code Number is used in the Service field of the Begin_Command of the Channel Protocol to specify the SAPI to which the channel is being established.)

3.2 Description of the Service

This specification defines the Service Access Protocol to access the HFEP implementation of the DoD standard Internet Protocol, MIL-STD-1777. The IP supports the interconnection of communication sub-networks within the Defense Data Network.

The IP Service Access Protocol provides the interface between a "client" process and the implementation of the IP in the HFEP configuration of the DDN Network Access Component. The phrase "client" process refers to a process operating within a subscriber device, e.g., a host computer system, that is connected to a NAC and supports communications via the HFP. The IP Service Access Protocol is implemented in both the client device and in the HFEP. Throughout this specification, "client process" will be used to refer to the portion of the SAP that is implemented in a subscriber device, and "service module" will be used to refer to the portion of the SAP that is implemented in the HFEP. The client process and the service module use the HFP Channel Protocol messages to exchange information. A synopsis of specific message types used follows.

begin command

> sent by the client process or the service module to initiate communications between the Host and the HFEP

begin response

sent by the client process or the service module to confirm the establishment of communications between the Host and the HFEP

end command

sent by the client process or the service module to terminate communications between the Host and the HFEP

end response

sent by the client process or the service module to confirm the termination of communications between the Host and the HFEP

execute command

not used

execute response

not used

transmit command

sent by the client process to transfer data to the service module for transmission to the remote IP

sent by the service module to transfer data received from the remote IP to the client process

The details of each message type are described in the following section.

3.3 Message Use

3.3.1 begin command

3.3.1.1 When Sent. A begin_command is sent by the client process to identify itself as an Internet ULP and to establish communications between the Host and the HFEP. An ULP is defined as any protocol above IP in the layered protocol hierarchy that uses IP. A

begin_command is sent by the service module to establish communications between the HFEP and the Host, in response to incoming data from the remote ULP.

3.3.1.2 Action on Receipt. When a begin_command is received by the client process or the service module, it should perform initialization processing to establish communications between the Host and the HFEP. Following completion of the processing, the client process or the service module should send a begin_response.

3.3.1.3 TEXT Field Syntax.

Protocol: FIXED(8)

3.3.1.4 TEXT Field Semantics.

a. Protocol. This field specifies the higher level protocol to be implemented, as described in MIL-STD-1777. Specific settings for this field are contained in ARPANET Request For Comments 870.[5] There is no default value for this field. If it is not set, the begin_response will return an error.

3.3.2 begin response

3.3.2.1 When Sent. The begin_response is sent by the client process or the service module to report the outcome of a begin_command.

3.3.2.2 Action on Receipt. When the service module or the client process receives a begin_response indicating successful initiation, it may proceed to transmit and receive data to and from the ULP. When the service module or the client process receives a begin_response indicating unsuccessful initiation, it should initiate appropriate error processing.

3.3.2.3 TEXT Field Syntax.

Channel Identifier: FIXED(16)

Result: FIXED(8)

3.3.2.4 TEXT Field Semantics.

a. Channel Identifier. This field will contain the HFP logical channel identifier to be used as the reference number in the subsequent messages.

b. Result. This field contains an encoding of the result of the begin_command. The codes are:

0 The command was successful.

1 A syntax error was detected in the TEXT field.

4 The local IP does not have sufficient resources to provide service.

12 The Protocol number specified is already in use.

3.3.3 end command

3.3.3.1 When Sent. The end_command is sent by the client process or the service module to request that communications be terminated between the Host and the HFEP. The end_command will be processed as a non-flushing termination by the Channel Protocol. (Note: non-flushing termination is specified by setting the Flush away bit to zero in the Control field. Termination mechanisms are described in the HFP Specification.)

3.3.3.2 Action on Receipt. When the client process or service module receives an end_command, it should perform termination processing to halt operation of the ULP. All data transmissions should be completed prior to ceasing operation. When the processing is completed, the client process or the service module should send an end_response.

3.3.3.3 TEXT Field Syntax.

Channel Identifier: FIXED(16)

3.3.3.4 TEXT Field Semantics.

a. Channel Identifier. This field specifies the HFP logical channel identifier for which operation should be terminated. There is no default value for this field.

3.3.4 end response

3.3.4.1 When Sent. The end_response is sent by the client process or the service module to confirm the termination of communications between the Host and the HFEP.

3.3.4.2 Action on Receipt. When the end_response is received by the client process or the service module, it should cease to operate as an ULP.

3.3.4.3 TEXT Field Syntax.

Channel Identifier: FIXED(16)
Result: FIXED(8)

3.3.4.4 TEXT Field Semantics.

a. Channel Identifier. The contents of this field will remain unchanged from the corresponding end_command.

b. Result. This field contains an encoding of the result of the end_command. The codes are:

0 The command was successful.

1 A syntax error was detected in the text field.

6 The logical channel that was specified to be terminated does not exist.

3.3.5 execute command

The execute_command is not used.

3.3.6 execute response

The execute_response is not used.

3.3.7 transmit command

3.3.7.1 When Sent. The transmit_command is sent by the client process to transfer data to the service module for transmission to the remote IP. The transmit_command is sent by the service module to transfer data to the client process that was received from the remote IP. Data may be sent only when the HFP channel is in the established state and when the HFP flow control discipline permits. (Note: Flow control mechanisms are described in the HFP specification.)

3.3.7.2 Action on Receipt. When a transmit_command is received by the client process, it should initiate processing of the data received from the remote IP. When a transmit_command is received by the service module, it should initiate the processing to transmit the data over the network.

3.3.7.3 TEXT Field Syntax.

Data Count:	FIXED(32)
Channel Identifier:	FIXED(16)
Data:	VARIABLE(?)

3.3.7.4 TEXT Field Semantics.

a. Data Count. This field specifies the size, in octets, of the data being transmitted. The default value is zero.

b. Channel Identifier. This field specifies the HFP logical channel identifier over which the data is being transmitted. There is no default value for this field.

c. Data. This field contains the IP options and data being transmitted. The format and definition of the Internet options is described in Paragraphs 9.3.13 and 9.3.15 of MIL-STD-1777.

REFERENCES

1. Day, J.D., et al., "WWMCCS Host to Front End Protocols: Specifications Version 1.0," DTI Document 78012.C-INFE.14, Digital Technology Incorporated, Champaign, IL, November 1979. (NTIS No. AD A100515/6)

2. "Derense Data Network Subscriber Interface Guide," Defense Communications Agency, Washington, D.C., July 1983.

3. MIL-STD-1778, Transmission Control Protocol, 12 August 1983.

4. MIL-STD-1/77, Internet Protocol, 12 August 1983.

5. Reynolds, J., and J. Postel, "Assigned Numbers," Network Working Group, RFC 870, October 1983.

DISTRIBUTION LIST

MITRE Washington

D-14 E. C. Brady
J. S. McManus
A. J. Roberts

W-30 W. B. Hall
J. S. Quilty

W-31 W. H. Blankertz
M. B. Charney
R. Coltun
H. C. Duffield
J. C. Hill
M. T. Louden
R. K. Miller (20)
J. R. Mullen
J. Nabielsky
S. S. Poh
R. W. Shirey
A. P. Skelton
D. C. Wood
Technical Staff

W-32 H. W. Neugent
H. I. Ottoson
A. M. Schoka
W. B. Stevens
J. M. Vasak

W-34 B. J. Moran
D. E. Zugby

W-35 C. E. Bowen
W. C. Kinzinger

W-36 G. W. Lipsey
J. C. Slaybaugh
M. J. Zobrak

W-37 R. C. Pesci
E. P. Schmidt

W-93 D. A. Whitaker

MITRE Washington Library

MITRE Bedford

D-34 S. J. Green
C. J. Murphy
B. P. Schanning

D-36 C. M. Sheehan

D-44 J. Mitchell

D-45 G. J. Koehr
M. A. Wingfield

MITRE Bedford Library

External

COL F. L. Maybaum, Code B610
Mr. J. Thomas, Code B611
Mr. M. Corrigan, Code B612
Mr. J. Powell, Code B613
LtCol J. Wegl, Code B615
Dr. T. Harris, Code B625
LTC W. Parrish, Code B626 (200)
Mr. R. Hyrkas, Code B626
Ms. J. Mallory, Code B626
Mr. W. Grindle, Code B627
Mr. R. Gutschmidt, Code B628

END

FILMED

5-85

DTIC